

日本国特許庁
PATENT OFFICE
JAPANESE GOVERNMENT

#3
Channeled
10-321



別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日

Date of Application:

2000年 6月21日

出願番号

Application Number:

特願2000-186410

出願人

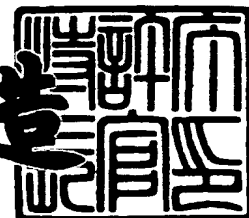
Applicant(s):

三菱電機株式会社

2000年 7月14日

特許庁長官
Commissioner,
Patent Office

及川耕造



出証番号 出証特2000-3055320

【書類名】 特許願

【整理番号】 523818JP01

【提出日】 平成12年 6月21日

【あて先】 特許庁長官殿

【国際特許分類】 G05B 19/00

【発明者】

 【住所又は居所】 東京都千代田区丸の内二丁目2番3号 三菱電機株式会社
社内

 【氏名】 仲井 勘

【発明者】

 【住所又は居所】 東京都千代田区丸の内二丁目2番3号 三菱電機株式会社
社内

 【氏名】 林 鋭志

【特許出願人】

 【識別番号】 000006013

 【住所又は居所】 東京都千代田区丸の内二丁目2番3号

 【氏名又は名称】 三菱電機株式会社

【代理人】

 【識別番号】 100062144

 【弁理士】

 【氏名又は名称】 青山 蓓

【選任した代理人】

 【識別番号】 100086405

 【弁理士】

 【氏名又は名称】 河宮 治

【手数料の表示】

 【予納台帳番号】 013262

 【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 デバイス制御プログラム開発手段及び実行手段

【特許請求の範囲】

【請求項 1】 1 つまたは複数の被制御対象オブジェクトを制御するプログラムを作成するプログラム開発手段と、上記の記述されたプログラムを実行するプログラム実行手段において、

上記プログラム開発手段及びプログラム実行手段にて使用されるソフトウェアにおいて上記被制御対象オブジェクトの各々に固有されるソフトウェアモジュールが独立して存在し、

上記ソフトウェアモジュールは、

- ・プログラム開発手段を構成する表示領域で表示される 1 つまたは複数のアイコンと、
 - ・上記被制御対象オブジェクトを制御するための方法を記述するための手続きと、
 - ・上記にて記述された被制御対象オブジェクトの制御方法を上記プログラム実行手段にて実行する手続きと
- のうちの、少なくとも 1 つを有し、

上記被制御対象オブジェクトソフトウェアモジュールはそのグローバル ID 若しくは相当のデータにより特定される、
プログラム開発手段及びプログラム実行手段。

【請求項 2】 1 つまたは複数のデバイスが被制御対象オブジェクトとして接続されている場合には、該デバイスからグローバル ID 若しくはそれに相当するデータが取得され、上記デバイスに対応する被制御対象オブジェクトソフトウェアモジュールはそのグローバル ID 若しくは相当のデータにより特定される、
請求項 1 に記載のプログラム開発手段及びプログラム実行手段。

【請求項 3】 上記デバイスがそれに係る被制御対象オブジェクトソフトウェアモジュールを備えており、
そのデバイスが接続され、
そのデバイスからそれに係る被制御対象オブジェクトソフトウェアモジュールが

取得される、

請求項 2 に記載のプログラム開発手段及びプログラム実行手段。

【請求項 4】 複数の被制御対象オブジェクトソフトウェアモジュールを備えるデータベースサーバから、

通信回線を介して、接続されているデバイス若しくは接続され得るデバイスを含む被制御対象オブジェクトに係る被制御対象オブジェクトソフトウェアモジュールが取得される、

請求項 2 に記載のプログラム開発手段及びプログラム実行手段。

【請求項 5】 上記プログラム開発手段が、接続されているデバイス若しくは接続され得るデバイスを含む被制御対象オブジェクトを、アイコンにより表示する被制御対象オブジェクト表示領域を備える、

請求項 2 に記載のプログラム開発手段及びプログラム実行手段。

【請求項 6】 上記被制御対象オブジェクトソフトウェアモジュールが、被制御対象オブジェクトの現在の状態に対応する複数種類のアイコンを有する、
請求項 5 に記載のプログラム開発手段及びプログラム実行手段。

【請求項 7】 上記プログラム開発手段が、プログラム作成領域を備え、
上記被制御対象オブジェクト表示領域に表示された被制御対象オブジェクトに係るアイコンを、該プログラム作成領域へ追加し、

追加されたそれら被制御対象オブジェクトに係るアイコンによりプログラムを構成する、

請求項 5 に記載のプログラム開発手段及びプログラム実行手段。

【請求項 8】 上記被制御対象オブジェクトソフトウェアモジュールが有する、上記被制御対象オブジェクトを制御するための方法を記述するための手続きを使って、該被制御対象オブジェクトの動作を設定していくことによりプログラムを構成する、

請求項 7 に記載のプログラム開発手段及びプログラム実行手段。

【請求項 9】 上記被制御対象オブジェクトソフトウェアモジュールが、被制御対象オブジェクトの設定動作に対応する複数種類のアイコンを有する、
請求項 8 に記載のプログラム開発手段及びプログラム実行手段。

【請求項10】 プログラム作成領域において、被制御対象オブジェクトアイコンを含む複数のアイコンを接続してフローチャートを形成することにより、プログラムの作成を行なう、

請求項7に記載のプログラム開発手段及びプログラム実行手段。

【請求項11】 上記被制御対象オブジェクト表示領域が、プログラムのシミュレーションを実行する際に、被制御対象オブジェクトの想定される状態に従って変動するアイコンを表示し、よって、上記被制御対象オブジェクト表示領域が、シミュレーション表示領域として利用される、

請求項9に記載のプログラム開発手段及びプログラム実行手段。

【請求項12】 上記プログラム実行手段が上記被制御対象オブジェクトを制御するプログラムを実行する際に、

上記被制御対象オブジェクト表示領域が、被制御対象オブジェクトの状態に従って変動するアイコンを表示し、よって、上記被制御対象オブジェクト表示領域が、モニタリング表示領域として利用される、

請求項6に記載のプログラム開発手段及びプログラム実行手段。

【請求項13】 上記プログラム実行手段にて稼動する、上記被制御対象オブジェクト制御プログラムが、グローバルIDを備えるデバイスを含む被制御対象オブジェクトに対して、メッセージを送信または受信するプログラム記述形態である、

請求項7に記載のプログラム開発手段及びプログラム実行手段。

【請求項14】 1つまたは複数のデバイスが被制御対象オブジェクトとして接続されている接続形態は、プラグ・アンド・プレイ機能又は活線抜粋をサポートするインタフェースであることを特徴とする、

請求項2に記載のプログラム開発手段及びプログラム実行手段。

【請求項15】 請求項7に記載のプログラム開発手段またはプログラム実行手段を構成し稼動させる、プログラムコードを、記録したことを特徴とする記録媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、自動機アプリケーションシステムを開発し実行するシステムに関する。

【0002】

【従来の技術】

以下のような自動機一般を構築する際には、例えばPLC (Programmable Logic Controller) やモーションコントローラなどのコントローラが利用される。

- ・ FA (Factory Automation) 分野で用いられる工作機械
- ・ 産業用ロボットなどの産業用自動機
- ・ 自律ロボット
- ・ 各種自動設備などの一般用途の自動機

それらコントローラには、コントローラに信号を入力するセンサやスイッチなどの各種入力デバイスが接続される。同様に、コントローラにより制御されるモータや表示器などの各種出力デバイスが接続される。また、コントローラには、それら入出力デバイスへの指令（指示）を記述するプログラムが搭載されている。プログラムに記述される指令（指示）が入出力デバイスに伝えられることにより、自動機は動作する。

【0003】

上記の入出力デバイスは、コントローラの入出力コネクタ（即ち、入出力ポート）に接続される。従来のコントローラを利用する場合に、プログラム作成者は、コントローラのどの（何番目の）入出力ポートにどの種類の入出力デバイスを接続するか（したか）を正確に把握する必要があった。その上で、コントローラ上で使用される、デバイスに係るソフトウェアモジュールや動作プログラムの設定が、正確に行われた。そうしなければ、コントローラが該デバイスに適切な指令（指示）を送って正しく制御することができないのである。なお、デバイスに係るソフトウェアモジュールには、例えばデバイスドライバが含まれる。

【0004】

パソコン（パーソナルコンピュータ）に対し拡張カードや周辺デバイスを追加

する場合にも、数年前までは上記と同様の事情が存在した。即ち、まずパソコン利用者は、装置本体の備える入出力ポートのどのものに、何の種類の周辺デバイスを接続するか（したか）を正確に把握する必要があった。その上で、パソコンが使用する、デバイスに係るソフトウェアモジュールや動作プログラムの設定が、正確に行われた。

【0005】

ところで、近年、パソコンとその周辺デバイスとの接続に関しては、プラグ・アンド・プレイという概念をもつインタフェースが開発された。そのインタフェースの利用により、パソコン利用者の上記のような負担が大幅に軽減されている。例えば、「USB」や「IEEE1394」のようなインタフェースがその例である。これらのインタフェースをサポートする周辺デバイスには、グローバルユニークID（GUID）と呼ばれる識別子が保持されている。

【0006】

GUIDは、各種デバイス別に（世界で）唯一割り当てられる識別子である。パソコンが、各種デバイスに各々対応するデバイスドライバなどのソフトウェアモジュールを予め相当量保持している場合を想定する。まず、入出力デバイスが接続されると該デバイスからGUIDが取得される。パソコンは、そのGUIDにより適切なソフトウェアモジュールを絞り込んで確定し、そして利用することができる。この機能によって、パソコン利用者が、デバイスドライバなどのソフトウェアモジュールの設定を行なうことの必要がなくなる。即ち、パソコン利用者はパソコン内にそのようなソフトウェアモジュールが存在していることすら意識することがない。にもかかわらず、デバイスを接続してデバイスを使用することが可能となる。但し、当該デバイスを使用する際のパラメータの調節が、パソコン利用者のなすべき作業として残っている。けれども、予め設定されている初期値をそのまま使用する限りこの作業は必要がない。

【0007】

USBやIEEE1394にはさらにホットプラグ（活線抜粋）の概念の機能も備わる。従来のパソコンでは、新たに接続されたデバイスに関するソフトウェアモジュールの設定を有効にするには、一度パソコンの電源を切り、そして再起

動する必要があった。上記のホットプラグ（活線抜粋）の機能は、パソコンの電源を切ること無しにデバイスを接続して使用することを可能とするものである。このホットプラグは、デバイス接続に関する設定での作業ををほぼ皆無にするものであり、パソコン使用者への負担を従来より大幅に軽減する。

【 0 0 0 8 】

USBやIEEE 1394といったプラグ・アンド・プレイの概念を持つインタフェースには、さらに別の特徴が備わる。それは、通信プロトコルが一般に公開されている（即ち、オープン性を有している）ことである。これにより、様々なサードパーティがパソコン周辺機器を作成することが可能である。また、従来、特定のデバイスをパソコンに接続するためには、特定の接続ボードなどの拡張ボードをパソコンに接続しなければならなかった。しかし、上記の特徴によりその煩わしさも無くなった。

【 0 0 0 9 】

このオープン性は、パソコンとその周辺デバイスにおいて実現されているものである。これに留まらずコントローラとその周辺デバイスの間の通信インタフェースにおいても利用されるニーズが高まっている。USBやIEEE 1394のような、オープン性を有するインタフェースの機能を備えるコントローラが試行されている。

【 0 0 1 0 】

ところで、上述のように、近年のコントローラ及びパソコンは、確かに、USBやIEEE 1394などのインタフェースを採用しつつある。しかしながら、それらのプラグ・アンド・プレイ機能が十分に生かされているとは言い難い状況にある。即ち、ただデバイスを接続するだけで即座にデバイスを使用することが可能となる、という状況は実現されていない。

【 0 0 1 1 】

まず、パソコンとその周辺デバイスについて説明する。例えばパソコンの一般的なマウスの場合、マウスを動かせばモニタ上の矢印が動くように設定されている。つまり、マウスがパソコンとの関係においてどのように動作するのかは予め決められ固定されている。あるいは、USBカメラを例に挙げる。USBカメラ

からの画像をパソコンディスプレイに表示するための特定のプログラムは、通常、USBカメラの製作者からUSBカメラと共に配布されている。しかも、そのプログラムに拠らなければ、そのUSBカメラは動作しない。即ち、パソコンに接続される周辺デバイスは、その用いられ方とパソコンとの関係における動作とが予め決められている。加えて、パソコン利用者がそのデバイスの用いられ方と動作とを自由に変更する（即ち、プログラムコードにより指示する）ことができない。

【 0 0 1 2 】

コントローラと周辺デバイスとについても説明する。そもそもコントローラにおいては、接続するデバイスの用いられ方をプログラムコードにより指示することが不可欠である。即ち、デバイスを接続すれば、即座にデバイスを使用することが可能になる、とはもともととなり得ない。必ずプログラムコードによるプログラミングが要求される。

【 0 0 1 3 】

しかも、コントローラ上で使用されるソフトウェアモジュールや動作プログラムの設定に関しては、従来のコントローラと同様に全て正確に行なう必要がある。USBやIEEE1394といったインタフェースを採用しているとはいえ、それらのプラグ・アンド・プレイ機能の利点が未だ十分には生かされていない。

【 0 0 1 4 】

【発明が解決しようとする課題】

本発明は、プラグ・アンド・プレイ機能を備えるインタフェースを用いるデバイスやコントローラによって自動機を構築する際に、利用されることを意図する。第1に、当該デバイスやコントローラに係るセットアップの手間を大幅に削減することを目的とする。第2に、シーケンスプログラム、イベントドリブンプログラム、モーションプログラムなどの各種制御プログラム作成を、簡素化することを目的とする。

【 0 0 1 5 】

【課題を解決するための手段】

本発明は、上記の目的を達成するためになされたものである。本発明に係る請

求項 1 に記載のプログラム開発手段及びプログラム実行手段は、

1 つまたは複数の被制御対象オブジェクトを制御するプログラムを作成するプログラム開発手段と、上記の記述されたプログラムを実行するプログラム実行手段である。

上記プログラム開発手段及びプログラム実行手段にて使用されるソフトウェアにおいて上記被制御対象オブジェクトの各々に固有されるソフトウェアモジュールが独立して存在し、

上記ソフトウェアモジュールは、

- ・プログラム開発手段を構成する表示領域で表示される 1 つまたは複数のアイコンと、
 - ・上記被制御対象オブジェクトを制御するための方法を記述するための手続きと、
 - ・上記にて記述された被制御対象オブジェクトの制御方法を上記プログラム実行手段にて実行する手続きと
- のうちの、少なくとも 1 つを有し、

上記被制御対象オブジェクトソフトウェアモジュールはそのグローバル ID 若しくは相当のデータにより特定される。

【 0 0 1 6 】

本発明に係る請求項 2 に記載のプログラム開発手段及びプログラム実行手段は、

1 つまたは複数のデバイスが被制御対象オブジェクトとして接続されている場合には、該デバイスからグローバル ID 若しくはそれに相当するデータが取得され、上記デバイスに対応する被制御対象オブジェクトソフトウェアモジュールはそのグローバル ID 若しくは相当のデータにより特定される、
請求項 1 に記載のプログラム開発手段及びプログラム実行手段である。

【 0 0 1 7 】

本発明に係る請求項 3 に記載のプログラム開発手段及びプログラム実行手段は、

上記デバイスがそれに係る被制御対象オブジェクトソフトウェアモジュールを

備えており、

そのデバイスが接続され、

そのデバイスからそれに係る被制御対象オブジェクトソフトウェアモジュールが
取得される、

請求項 2 に記載のプログラム開発手段及びプログラム実行手段である。

【 0 0 1 8 】

本発明に係る請求項 4 に記載のプログラム開発手段及びプログラム実行手段は

、
複数の被制御対象オブジェクトソフトウェアモジュールを備えるデータベース
サーバから、

通信回線を介して、接続されているデバイス若しくは接続され得るデバイス
を含む被制御対象オブジェクトに係る被制御対象オブジェクトソフトウェアモ
ジュールが取得される、

請求項 2 に記載のプログラム開発手段及びプログラム実行手段である。

【 0 0 1 9 】

本発明に係る請求項 5 に記載のプログラム開発手段及びプログラム実行手段は

、
上記プログラム開発手段が、接続されているデバイス若しくは接続され得るデ
バイスを含む被制御対象オブジェクトを、アイコンにより表示する被制御対象オ
ブジェクト表示領域を備える、

請求項 2 に記載のプログラム開発手段及びプログラム実行手段である。

【 0 0 2 0 】

本発明に係る請求項 6 に記載のプログラム開発手段及びプログラム実行手段は

、
上記被制御対象オブジェクトソフトウェアモジュールが、被制御対象オブジェ
クトの現在の状態に対応する複数種類のアイコンを有する、

請求項 5 に記載のプログラム開発手段及びプログラム実行手段である。

【 0 0 2 1 】

本発明に係る請求項 7 に記載のプログラム開発手段及びプログラム実行手段は

上記プログラム開発手段が、プログラム作成領域を備え、

上記被制御対象オブジェクト表示領域に表示された被制御対象オブジェクトに係るアイコンを、該プログラム作成領域へ追加し、

追加されたそれら被制御対象オブジェクトに係るアイコンによりプログラムを構成する、

請求項 5 に記載のプログラム開発手段及びプログラム実行手段である。

【 0 0 2 2 】

本発明に係る請求項 8 に記載のプログラム開発手段及びプログラム実行手段は

上記被制御対象オブジェクトソフトウェアモジュールが有する、上記被制御対象オブジェクトを制御するための方法を記述するための手続きを使って、該被制御対象オブジェクトの動作を設定していくことによりプログラムを構成する、

請求項 7 に記載のプログラム開発手段及びプログラム実行手段である。

【 0 0 2 3 】

本発明に係る請求項 9 に記載のプログラム開発手段及びプログラム実行手段は

上記被制御対象オブジェクトソフトウェアモジュールが、被制御対象オブジェクトの設定動作に対応する複数種類のアイコンを有する、

請求項 8 に記載のプログラム開発手段及びプログラム実行手段である。

【 0 0 2 4 】

本発明に係る請求項 1 0 に記載のプログラム開発手段及びプログラム実行手段は、

プログラム作成領域において、被制御対象オブジェクトアイコンを含む複数のアイコンを接続してフローチャートを形成することにより、プログラムの作成を行なう、

請求項 7 に記載のプログラム開発手段及びプログラム実行手段である。

【 0 0 2 5 】

本発明に係る請求項 1 1 に記載のプログラム開発手段及びプログラム実行手段

は、

上記被制御対象オブジェクト表示領域が、プログラムのシミュレーションを実行する際に、被制御対象オブジェクトの想定される状態に従って変動するアイコンを表示し、よって、上記被制御対象オブジェクト表示領域が、シミュレーション表示領域として利用される、

請求項 9 に記載のプログラム開発手段及びプログラム実行手段である。

【 0 0 2 6 】

本発明に係る請求項 1 2 に記載のプログラム開発手段及びプログラム実行手段は、

上記プログラム実行手段が上記被制御対象オブジェクトを制御するプログラムを実行する際に、

上記被制御対象オブジェクト表示領域が、被制御対象オブジェクトの状態に従って変動するアイコンを表示し、よって、上記被制御対象オブジェクト表示領域が、モニタリング表示領域として利用される、

請求項 6 に記載のプログラム開発手段及びプログラム実行手段である。

【 0 0 2 7 】

本発明に係る請求項 1 3 に記載のプログラム開発手段及びプログラム実行手段は、

上記プログラム実行手段にて稼動する、上記被制御対象オブジェクト制御プログラムが、グローバル ID を備えるデバイスを含む被制御対象オブジェクトに対して、メッセージを送信または受信するプログラム記述形態である、

請求項 7 に記載のプログラム開発手段及びプログラム実行手段である。

【 0 0 2 8 】

本発明に係る請求項 1 4 に記載のプログラム開発手段及びプログラム実行手段は、

1 つまたは複数のデバイスが被制御対象オブジェクトとして接続されている接続形態は、プラグ・アンド・プレイ機能又は活線抜粋をサポートするインタフェースであることを特徴とする、

請求項 2 に記載のプログラム開発手段及びプログラム実行手段である。

【 0 0 2 9 】

本発明に係る請求項 1 5 に記載の記録媒体は、
請求項 7 に記載のプログラム開発手段またはプログラム実行手段を構成し稼動させる、プログラムコードを、記録したことを特徴とする記録媒体である。

【 0 0 3 0 】

【発明の実施の形態】

以下、図面を参照しつつ、本発明に係る好適な実施の形態について説明する。

【 0 0 3 1 】

実施の形態 1 .

図 1 は、本発明の実施の形態 1 に従い、自動機アプリケーションシステムを構築する手順の一例を示したものであり、以下この手順に従って説明を行う。

【 0 0 3 2 】

アプリケーションを構築する際には、まず、

①デバイスをコントローラに接続して自動機を組み立てる（図 1 ・ S 0 2 ）。

【 0 0 3 3 】

図 2 （ 1 ）は、コントローラ 6 と周辺デバイス 1 0 とを接続する一例を示す。
プログラム作成手段 2 とコントローラ 6 とは、第 1 の接続手段 4 により接続される。コントローラ 6 と周辺デバイス 1 0 とは、第 2 の接続手段 8 とにより接続される。プログラム作成手段 2 は、一般的にはパソコン（パーソナル・コンピュータ）である。オペレーティングシステムとしては、一般的には M i c r o s o f t （マイクロソフト）社の「W i n d o w s 9 8」、「W i n d o w s N T」若しくは「W i n d o w s 2 0 0 0」が利用されるが、アップルコンピュータ社の「M a c O S」や、更には L i n u x などでもよい。またプログラム作成手段 2 は、テレビジョン受像機と接続して用いるゲーム機や、表示画面を備えて単体で用いるゲーム機（例えば、携帯用ゲーム機）などであってもよい。

【 0 0 3 4 】

また、プログラム作成手段 2 がコントローラ 6 と一体であってもよい。図 2 （ 2 ）はその一例であって、コントローラ 6 がプログラム作成手段 2 の機能を有する。この例では、コントローラ 6 は、プログラムを作成する際に使用する表示画

面や設定ボタンを有しているのが好ましい。その表示画面がタッチパネル等の機能を備える場合には設定ボタンは備わらなくてもよい。図 2 (3) は、プログラム作成手段 2 がコントローラ 6 と一体である別の例であって、プログラム作成手段 2 がコントローラ 6 の機能を有する。

【 0 0 3 5 】

第 1 の接続手段 4、第 2 の接続手段 8 は、同じ接続形態でもそれぞれ異なる接続形態でもよい。また、USB や IEEE 1 3 9 4 等の有線でも Bluetooth や IrDA 等の無線でもよい。

【 0 0 3 6 】

次に、②接続されている周辺デバイス 1 0 の情報を、第 1 の接続手段 4 及び第 2 の接続手段 8 を介して、コントローラ 6 から取得する (図 1・S 0 4)。各デバイス 1 0 は、そのデバイス 1 0 の製作者 (製作会社) 及び機種を識別するための、GUID (グローバルユニーク ID) と呼ばれる識別子を有する。識別子に相当するデータを有する形態でもよい。それら識別子 (または識別子相当のデータ) は、メモリ部 (図示せず。) に格納される。プログラム作成手段 2 上で稼動するプログラム開発環境 2 0 は、GUID を収集する。このことにより、コントローラ 6 に接続されている周辺デバイス 1 0 が何であるかを把握することができる。

【 0 0 3 7 】

次に、③接続されている周辺デバイス 1 0 に対応するデバイスドライバをリンク (関連付け) する (図 1・S 0 6)。プログラム開発環境 2 0 は、利用し得る全てのデバイスに対応するデバイスドライバ群を有したデータベースにアクセスする。そして、GUID に対応するデバイス 1 0 のデバイスドライバに対してリンクを張る。または、当該デバイスドライバをプログラム開発環境 2 0 内に持ってくる (即ち、例えば、コピーする、若しくはダウンロードする)。デバイスドライバは DLL (ダイナミック・リンク・ライブラリ) の形態で提供されても構わない。

【 0 0 3 8 】

図 3 は、本発明の実施の形態 1 に係るソフトウェアアーキテクチャの一例を示

した図である。該アーキテクチャには、プログラム開発環境 2 0、プログラム実行環境 2 2、及び、デバイス 1 0 に対応するデバイスドライバ 2 4 が含まれる。図 3 は、コントローラ 6 が、プログラム作成手段 2 の機能を有する場合を示すものである。コントローラ 6 とプログラム作成手段 2 とが、別々に存在する場合も、もちろん想定され得る。この場合、プログラム作成手段 2 がプログラム開発環境 2 0 を有し、コントローラ 6 がプログラム実行環境 2 2 を有する。プログラム作成手段 2 は、プログラムを開発する際にシミュレーションを行なうため、プログラム実行環境 2 2 も同時に有しても構わない。

【 0 0 3 9 】

デバイスドライバ 2 4 は、以下のモジュールのうち少なくともいずれか一つを有している。

- ・プログラム開発環境 2 0 で表示されるアイコン
- ・当該デバイスを制御するための方法を記述するための手続き
- ・上記の当該デバイスの制御方法をプログラム実行環境 2 2 上で実行する手続き
- ・当該デバイスと通信するための通信手続き

図 4 (1) は、異なるデバイス 1 0 を接続する場合に想定されるソフトウェアアーキテクチャである。

【 0 0 4 0 】

プログラム実行環境 2 2 が、通信手続き 2 6 を提供する場合もある。異なるデバイス 1 0 でも同じ通信手段を使用している場合には、共通の通信手続き 2 6 を利用することになる。図 4 (2) はその様子を示す。同じ通信手段とは、例えばどちらも U S B デバイスであるような場合である。一方、異なるデバイス 1 0 で異なる通信手段を使用している場合には、異なる通信手続き 2 6 を利用することになる。図 4 (3) はその様子を示す。異なる通信手段とは、例えば一方が U S B デバイスで他方が I E E E 1 3 9 4 デバイスであるような場合である。

【 0 0 4 1 】

利用し得る全てのデバイス 1 0 に対応するデバイスドライバ群を有したデータベースは、プログラム開発環境 2 0 内に存在してもよい。図 5 (1) はその例を示し、データベースはパソコン 2 内のハードディスク 4 0 上に存在する。また、

プログラム開発環境 2 0 外の外部記憶媒体 4 2 に存在してもよい。図 5 (2) はその例を示し、データベースは、パソコン 2 でアクセス可能な記憶媒体 (C D - R O M 、 D V D - R O M 、 ゲームソフト記憶手段、等) 上に存在する。さらに、遠隔地のサーバ内 (例えば、図 5 (3) のようにサーバのデータベース記憶領域 4 4) に存在してもよい。この場合には図 5 (3) に示されるように、L A N やインターネットなどの伝送経路 4 6 を通じてアクセスすることになる。

【 0 0 4 2 】

各周辺デバイス 1 0 が、自らのデバイスドライバ 2 4 を提供するようにしてもよい。即ち、まず、例えば各デバイス 1 0 が、そのデバイスが何であることを識別するための G U I D のみならずデバイスドライバ 2 4 も備えている。そこで、プログラム開発環境 2 0 は接続されているデバイス 1 0 から直接デバイスドライバ 2 4 を取得してもよい。この場合、デバイスドライバ群を有したデータベースにアクセスする必要はない。但し、デバイス 1 0 における記憶領域では、G U I D を記憶する領域容量に加えてデバイスドライバ 2 4 を記憶する分だけ大きい領域容量が必要となる。

【 0 0 4 3 】

次に、④プログラム開発環境 2 0 のデバイス表示ウィンドウ (領域) 6 6 上にデバイスアイコン 6 8 を表示する (図 1 ・ S 0 8) 。アイコンを有しているデバイスドライバ 2 4 に関しては、当該アイコンを表示する。

【 0 0 4 4 】

図 6 は、本発明に係るプログラム開発環境 2 0 の一例を示した図である。該プログラム開発環境 2 0 は、以下の要素を含んでいる。

- ・プログラム開発環境画面全体 6 0
- ・プログラム開発環境 2 0 の基本的な操作を取り扱うツールバー 6 2
- ・プログラム作成領域 6 4
- ・デバイス表示領域 6 6

プログラム作成領域 6 4 内に、アイコンをドラッグアンドドロップしたり、アイコン同士を接続したりすることで、プログラムを作成することを想定している。デバイス表示領域 6 6 は、構築しようとする自動機アプリケーションシステムで

使用するデバイス10をアイコン形式で表示する領域である。該領域66の形状は、図6(1)のように、ウインドウ形式でもよく、図6(2)のように、ツールバーの形式でもよく、更に図6(3)のように、ツリービュー形式でもよい。

【0045】

次に、⑤デバイスアイコン68を、デバイス表示領域66から、プログラム表示領域64へドラッグ・アンド・ドロップする。即ち、デバイス表示領域66に表示されているデバイスアイコン68を、プログラム作成領域64上に新たに作成する(図1・S10)。プログラム開発環境20は、プログラム作成に必要な種々の制御コマンド(分岐コマンド、繰り返しコマンド、若しくは割込みコマンド等)をアイコン形式で表示する制御コマンド表示領域(図示せず。)を備えているのが好ましい。必要な制御コマンドのアイコンも、プログラム作成領域64へ、ドラッグアンドドロップする(新たに作成する)。

【0046】

次に、⑥プログラム作成領域64上のアイコン同士を接続し、コントローラ6が実行する制御プログラムを作成する(図1・S12)。アイコン同士の接続においては、アイコンとアイコンを隣り合わせに接続する方法によってもよい。アイコン同士を線で接続するフローチャート形式による方法でもよい。

【0047】

図7(1)は、本発明に係るプログラム開発の一例を示した図である。図7(1)におけるプログラム開発環境画面60では、以下の要素が含まれる。

- ・デバイス表示領域66からプログラム作成領域64へドラッグ・アンド・ドロップされたデバイスアイコン68'
- ・プログラムの開始を示している制御コマンドアイコンである開始アイコン70
- ・開始アイコン70とデバイスアイコン68'とをフローチャート形式に則り接続する矢印72
- ・デバイスアイコン68'の動作を記述するためのダイアログボックス(またはアプリケーションソフトウェア)88

【0048】

ダイアログボックス88は、デバイスアイコン68'がダブルクリックされる

ことにより開かれる。デバイスドライバ24が該デバイス10を制御するための方法を記述するための手続きを備える場合は、その手続きに従ったダイアログボックスが提示される。デバイス表示領域66からプログラム作成領域64へドラッグ・アンド・ドロップされた複数のデバイスアイコン68'をプログラミングしてから（即ち、デバイスアイコン68'同士を接続してから）、各アイコン68'に対してダイアログボックス88を開いて動作記述をしてもよい。各アイコン68'に対してダイアログボックス88を開いて動作記述をしてから、アイコン68'同士を接続してもよい。順序はどのようなでも構わない。デバイスドライバ24が、デバイス10の設定動作や現在の状況に応じた複数種類のアイコンを有している場合は、その動作設定に従ったアイコンが表示される（例えば、図7（2）のアイコン68''）。

【0049】

上記のデバイスアイコン68'の動作の記述において、アイコン動作設定用のアプリケーションソフトウェアを立ち上げて動作を設定する、というように構成することも可能である。この場合、より複雑な動作を記述することができる。

【0050】

最後に、⑦作成されたプログラムとデバイスドライバ24の必要部分とを、プログラム実行環境22（コントローラ6）へ転送し、そして実行する（図1・S14）。例えば、プログラム実行環境22が通信手続きを提供する場合には、通信手続きに関連する部分は転送される必要はない。従って、デバイスドライバ24に関しては、必要部分のみ転送される。

【0051】

また、コントローラ6がプログラムの進行状況を表示するための表示手段を備えていることがある。このとき、アイコン68も転送すれば、デバイスの状態に応じたアイコン68、68'、68''をコントローラ6の表示手段に表示することができる。

【0052】

以上の実施の形態1に係るプログラム開発環境20及びプログラム実行環境22の動作は、コンピュータ制御に適うプログラムコードにより記述されるもので

ある。これらプログラムコードが、パソコン（等）のメモリ部（図示せず。）や処理部（図示せず。）にて、格納され稼動制御される。これらプログラムは、CD-ROMなどの記録媒体に記録することができる。

【 0 0 5 3 】

実施の形態 1 を利用することにより、以下のような効果が得られる。

【 0 0 5 4 】

接続されているデバイス 1 0 に対応すべきデバイスドライバ 2 4 を自動的にリンクする。従って、制御プログラム作成においては、当該デバイス 1 0 の動作に関して必要最小限の記述をするだけでよい。即ち、当該デバイス 1 0 に係るその他の設定を一切する必要がない。よって、プラグ・アンド・プレイ機能が十分に生かされ制御プログラムの開発効率が格段に向上する。

【 0 0 5 5 】

プログラム可能なデバイス 1 0 のみが利用可能であるからプログラム作成のミスを未然に防げることが、挙げられる。当該デバイス 1 0 に係るソフトウェアの大半は、デバイスドライバ 2 4 内部にカプセル化されている。従って、それら部分のコーディング内容に関する知識が無くとも動作実証済みのデバイスドライバ 2 4 にリンクすることで大規模なアプリケーションを構築し得る。該構築者が理解する必要があるのは、デバイスドライバ 2 4 によって提供される機能と、該デバイスドライバ 2 4 が使用する外部パラメータのみである。制御プログラム生成者は、外部パラメータの値しか変更できない。よって、整合性のない矛盾したデータ型のデータや制御信号を利用する、というような危険性は大幅に減少するのである。

【 0 0 5 6 】

また、アイコンのドラッグ・アンド・ドロップと、アイコン同士の接続という基礎的な操作のみでプログラム作成が可能である。プログラムは、フローチャート形式に則り作成される。プログラム作成が容易であるだけでなく、作成されたプログラムの可読性もよい。デバイスが、設定や状況に応じて複数種類のアイコンを表示することも、作成されたプログラムの可読性がよいことに貢献する。

【 0 0 5 7 】

実施の形態 2.

図 8 は、本発明の実施の形態 2 に従い、仮想的に自動機アプリケーションシステムを構築する手順の一例を示したものである。以下、この手順に従って説明を行なう。

【0058】

仮想アプリケーションを構築する際には、まず、

①自動機をデバイス表示領域 6 6 上で組み立てる。即ち、デバイスアイコン 6 8 を追加する (図 8・S102)。

【0059】

図 9 は、本発明に係るプログラム開発環境 2 0 の一例を示した図である。該プログラム開発環境 2 0 は、以下の要素を含んでいる。

- ・プログラム開発環境画面全体 6 0
- ・プログラム開発環境 2 0 の基本的な操作を取り扱うツールバー 6 2
- ・プログラム作成領域 6 4
- ・デバイス表示領域 6 6

このプログラム開発環境 2 0 は、前に説明した図 6 (1) と同じであっても構わない。プログラム開発環境 2 0 は、利用し得る全てのデバイス 1 0 を (図 9 (1) ではダイアログボックス 1 0 0 によって) 提示する。その中から所望のデバイス 1 0 を選択し、デバイスアイコン 6 8 をデバイス表示領域 6 6 に追加する。利用し得る全てのデバイス 1 0 に対応するデバイスドライバ群を有したデータベースは、前に説明した図 5 のものと同じであってよい。

【0060】

仮想的に自動機アプリケーションシステムを構築するに際しては、このようにデバイスアイコン 6 8 をデバイス表示領域 6 6 に追加するだけでもよい。どのような自動機アプリケーションシステムを構築するのかを分かりやすくするために、「絵」を書き加えてもよい。図 9 (2) はその一例で、符号 1 0 2 により示されるのが、書き加えられた絵である。

【0061】

②デバイス 1 0 に対応するデバイスドライバ 2 4 は、デバイスアイコン 6 8 を

デバイス表示領域 6 6 へ追加する際に自動的にリンクされる (図 8・S 1 0 4)。
実施の形態 1 では、GUID を取得してコントローラ 6 に接続されているデバイス 1 0 が何であるかを判断する。一方、実施の形態 2 では、利用し得る全てのデバイス 1 0 を画面 6 0 にて提示し、その中から所望のデバイス 1 0 を選択する。
よって、その選択により、実施の形態 1 と同様に、当該デバイスドライバ 2 4 のリンクを行うことができる。

【 0 0 6 2 】

次に、③動作させるアイコン 6 8 を、プログラム作成ウインドウ (領域) 6 4 へドラッグ・アンド・ドロップする (図 8・S 1 0 6)。続いて、
④プログラム作成領域 6 4 上のアイコン同士を接続し、コントローラ 6 が実行する制御プログラムを作成する (図 8・S 1 0 8)。
このようなプログラムの作成も、実施の形態 1 と同じである。

【 0 0 6 3 】

⑤シミュレーションにより作成されたプログラムの動作を確認する (図 8・S 1 1 0)。図 1 0 (1) のプログラム作成領域 6 4 は、作成されたプログラムの例を示す。作成したプログラムの動作を確認するためのシミュレーションを、プログラム開発環境 2 0 上で実行することができる。シミュレーションを実行すると、プログラム作成領域 6 4 では、動作プログラム通りに実行していく。その際、プログラム上で現在どのブロック (即ち、アイコン、コマンド) を実行しているのかを表示する。一方デバイス表示領域 6 6 上のデバイスアイコン 6 8 は、そのデバイス 1 0 のシミュレーション時の現況を示している。デバイスドライバ 2 4 が、デバイス 1 0 の設定動作や現在の状況に応じた複数種類のアイコンを有している場合は、その動作設定に従ったアイコンが表示される。図 1 0 (2) はその 1 例である。プログラム上では ON の所を実行しており、デバイス表示領域 6 6 にあるデバイスアイコン 6 8 は、「ON」された状態を示している。

【 0 0 6 4 】

ここで作業を終了し、作成したプログラムを保存することもできる。

【 0 0 6 5 】

また、⑥作成されたプログラムとデバイス 1 0 のソフトウェアモジュールをと

コントローラ 6 へ転送し実行することもできる（図 8・S112）。

【0066】

この場合、以下の 2 者は同一でなければならない。

(1) プログラム開発環境 20 上で仮想的に構築したアプリケーションの構成

(2) 実際のコントローラ 6 とデバイス 10 とを接続して実際に構築したアプリケーションの構成

但し、これを確認することは容易である。即ち、作成されたプログラムとデバイス 10 のソフトウェアモジュールをコントローラ 6 へ転送する前に、接続されるデバイス 10 の GUID を収集すればよい。そうすると、仮想的に構築したアプリケーションのデバイス情報と、実際の GUID により得られる情報とを、容易に比較することができる。もし両者が異なるならば、その旨を示す警告画面をユーザに示す等の処置をとることができる。

【0067】

たとえ、上記確認作業を行わずに、作成されたプログラムとデバイス 10 のソフトウェアモジュールとをコントローラ 6 へ転送し実行しても、問題は生じない。コントローラ 6 上の制御プログラムは、動作させようとするデバイス 10 と実際に接続されるデバイス 10 とが、一致する場合にのみ機能し得る。所望のデバイスが実際に接続されていない場合には、制御プログラムによる動作に係る指令（指示）は無視される。よって、予期せぬ事態が起こるようなことはなく、フェールセーフである。

【0068】

従って、制御プログラムに記述されていないデバイス 10 が実際にコントローラにつながっていたとしても、同様にフェールセーフとなり問題は起こらない。

【0069】

シミュレーションで行なった内容は、実際のコントローラ 6 の稼動状況のモニタリングにも使用することができる。即ち、シミュレーション画面はそのままモニタリング画面として、利用することができる。

【0070】

以上の実施の形態 2 に係るプログラム開発環境 20 及びプログラム実行環境 2

2の動作は、コンピュータ制御に適うプログラムコードにより記述されるものである。これらプログラムコードが、パソコン（等）のメモリ部（図示せず。）や処理部（図示せず。）にて、格納され稼動制御される。これらプログラムは、CD-ROMなどの記録媒体に記録することができる。

【 0 0 7 1 】

実施の形態1では、実際に自動機アプリケーションシステムを構築する例を示した。一方、実施の形態2では、仮想的に自動機アプリケーションシステムを構築する例を示した。両者を混合する形態も、実施可能である。

【 0 0 7 2 】

実施の形態2を利用することにより、以下のような効果が得られる。

【 0 0 7 3 】

実際にデバイス10をコントローラ6やプログラム作成手段2に接続しなくとも（即ち、実際に手元にデバイス10がなくとも）、そのデバイス10に係るプログラムを作成することができる。仮想的なアプリケーション構築やプログラム作成においても、接続されるはずのデバイス10に対応するデバイスドライバ24は自動的にリンクされる。従って、当該デバイス10の動作に関して必要最小限の記述をするだけでよい。即ち、当該デバイス10に係るその他の設定を一切する必要がない。よって、仮想的にプラグ・アンド・プレイ機能が十分に生かされ制御プログラムの開発効率が格段に向上する。

【 0 0 7 4 】

さらには、シミュレーションによりプログラム作成のミスを未然に防ぐことができる。プログラムはフローチャート形式で記述され、且つ現在のシミュレーション状況がフローチャートに従い逐次表示されるので、シミュレーションを理解しやすい。上述のように、デバイスドライバ24が、デバイス10の設定動作や現在の状況に応じた複数種類のアイコンを有していることがある。その場合には、その動作設定に従ったアイコンが表示され得るため、仮想的に構築したアプリケーションの動作の状況が分かりやすく表示される。また、上述のように、どのような自動機アプリケーションシステムを構築するのかを分かりやすくするために絵を書き加えることができる。このことにより、仮想的に構築したアプリケー

ションシステムの動作の状況がさらにに分かりやすく表示され得る。

【 0 0 7 5 】

シミュレーションで行なった内容は、実際のコントローラ 6 の稼動状況のモニタリングにおいても、使用することができる。即ちシミュレーション画面はそのままモニタリング画面として、利用することができる。制御プログラムの開発におけるモニタリング画面構築の作業を省くことができる。

【 0 0 7 6 】

本発明では、実際に手元にデバイス 1 0 が無くとも当該デバイス 1 0 に係わる動作プログラムを記述することができる。この発明を利用すれば、大規模アプリケーションの動作プログラムの開発を行なう場合に、動作プログラムを分割して作成してゆくことが可能である。

【 0 0 7 7 】

実施の形態 3.

本発明の実施の形態 3 は、被制御対象オブジェクトがデバイス 1 0 として実在せずに、モニタリング画面に所謂「バーチャルアシスタント (Virtual Assistant)」の形態として存在するような場合の、一つの例である。ここで、「被制御対象オブジェクト」とは、自動機アプリケーションシステムが制御の対象として認識するオブジェクトである。実存在であるデバイス 1 0 や仮想的存在であるバーチャルアシスタントが、この「被制御対象オブジェクト」に含まれる。バーチャルアシスタント (以下では、バーチャル管理人と称する。) 7 4 は、モニタリング画面上にのみ存在する。

【 0 0 7 8 】

本発明の実施の形態 3 に従い、仮想的に自動機アプリケーションシステムを構築する手順は、図 8 の実施の形態 2 に係る手順と同じでよい。

【 0 0 7 9 】

図 1 1 は、本発明に係るプログラム開発環境 2 0 の一例を示した図である。

【 0 0 8 0 】

自動機アプリケーションシステムを構築するに際しては、モニタリング画面の構築もまた重要な位置を占める。よって、モニタリング画面においては、自動機

アプリケーションシステムで実際に使用されるデバイス 10 の現況をモニタリングすることのみに留まらないことが好ましい。即ち、操作者にモニタリング現況の内容をよりわかりやすく示すために、上記の「バーチャル管理人」74 をモニタリング画面に登場させることが好ましい。

【 0 0 8 1 】

実施の形態 2 の説明で述べたように、シミュレーション画面はそのままモニタリング画面として利用することができる。実施の形態 2 のプログラム開発環境画面 60 では、自動機アプリケーションシステムで実際に使用されるデバイス（被制御対象オブジェクト）10（に係るデバイスアイコン 68）を、デバイス表示領域 60 及びプログラム作成領域 64 に配置する。シミュレーション時及びモニタリング時には、そのプログラム作成領域 64 及びデバイス表示領域 64 にてデバイス 10 の動作がデバイスアイコン 68'、68 により示される。以上と同様に、バーチャル管理人 74 をプログラム開発環境画面 60 のデバイス表示領域 66 及びプログラム作成領域 64 にて配置する。そうすれば、モニタリング画面でのみ動作するバーチャル管理人 74' を作成し利用することができる。

【 0 0 8 2 】

図 12 は、バーチャル管理人 74 に関連するプログラムの開発の一つの例を示す図である。図 12 において、条件分岐 76 のブロックが利用されている。ダイアログボックス 78 により、被制御対象オブジェクト（図 12 では、バーチャル管理人）74' の動作の記述を行なう。ダイアログボックス 78 は、例えば、被制御対象オブジェクト（バーチャル管理人）74' がダブルクリックされることにより開かれる。実施の形態 1 での、デバイスアイコン 68' の動作を記述するダイアログボックス 88 は、所謂「チェック形式」により動作が設定される。一方、図 12（1）では、バーチャルアシスタント 74' の動作を記述するダイアログボックス 78 は、所謂「リスト形式」により動作が設定されている。勿論、どちらの形式による動作設定であってもよく、また別の形式であってもよい。

【 0 0 8 3 】

実施の形態 1 と同じように、バーチャル管理人 78' の動作記述において、アイコン動作設定用のアプリケーションソフトウェアを立ち上げて動作を設定する

、というように構成することも可能である。この場合、より複雑な動作を記述することができる。

【 0 0 8 4 】

作成されたプログラムは、コントローラ 6 へ転送される。コントローラ 6 が表示手段を備えない場合（例えば、図 2（2））、バーチャル管理人 7 4 に係る部分は、例えば、無視されることになる。このとき、コントローラ 6 にはプログラム全てが転送され、実行時にコントローラ 6 が無視するという形態でもよい。また、プログラムを転送する時点で、バーチャル管理人 7 4 に係るプログラムの部分が除かれて、コントローラ 6 に送られてもよい。

【 0 0 8 5 】

バーチャル管理人（バーチャルアシスタント） 7 4 に対応する実在のデバイス 1 0 は存在しない。但し、実在するデバイス 1 0 が有するグローバルユニーク ID に相当する ID コードを、バーチャル管理人（バーチャルアシスタント） 7 4 に備えさせることが可能である。このとき、例えば、現実のデバイス 1 0 に割り当てられている GUID とは全く異なるコードを、上記の「相当する ID コード」として設定すればよい。すると、コントローラ 6 にどんなデバイス 1 0 が接続されても、それらのデバイス 1 0 に対し、バーチャル管理人（バーチャルアシスタント） 7 4 に関連する動作の指示が与えられることは決して無い。よって、予期せぬ事態が起こるようなことは無く、フェールセーフである。

【 0 0 8 6 】

図 1 2（2）は、作成されたプログラムを実行したときの一例である。この例では、開始後、条件分岐 7 6 により、“ON”を実行するフローへは進まないで、「バーチャル管理人 7 4 が右手を挙げる」フローに進んでいる、という状況を示している。即ち、図示されるように、デバイス表示領域 6 6 上のバーチャル管理人 7 4 は、右手を挙げている。

【 0 0 8 7 】

以上の実施の形態 3 に係るプログラム開発環境 2 0 及びプログラム実行環境 2 2 の動作は、コンピュータ制御に適うプログラムコードにより記述されるものである。これらプログラムコードが、パソコン（等）のメモリ部（図示せず。）や

処理部（図示せず。）にて、格納され稼動制御される。これらプログラムは、C D - R O M などの記録媒体に記録することができる。

【 0 0 8 8 】

実施の形態 3 を利用することにより、以下のような効果が得られる。

【 0 0 8 9 】

モニタリング画面にてより多くの情報を提示することが可能になる。しかもバーチャルアシスタント 7 4 は独自の形状（例えば「人」形状）を有するアイコン（等）により表現されているので、前記情報が分かりやすい形で操作者に与えられる。

【 0 0 9 0 】

以下には、実施の形態 1、実施の形態 2 及び実施の形態 3 に共通の効果を示す。

【 0 0 9 1 】

I E E E 1 3 9 4 は、所謂「ホームネットワーク」において利用し得るインタフェースである。従って、本発明の実施の形態にて I E E E 1 3 9 4 を利用すれば、デバイス 1 0 として冷蔵庫、電子レンジ、エアコン、若しくは照明器具などの家電製品を接続することができる。このときコントローラ 6 には、タイマ動作などをプログラミングしておくことができる。

【 0 0 9 2 】

また、本発明の利用により、特にファクトリオートメーションなどの自動機分野において、ホットプラグ機能を充分に活用することが可能になる。例えば、ある自動機に既にプログラムされ実行されているアプリケーションシステムに対し、デバイス 1 0 を追加若しくは削除する（引き抜く）場合を想定する。この場合従来であれば、コントローラ 6 の電源を一旦 O F F にして、プログラムを追加若しくは削除等の変更をする必要があった。ホットプラグ機能を利用して本発明を実施することにより、電源 O N のままプログラムの追加や削除等の変更をし、かつデバイス 1 0 を追加若しくは削除する（引き抜く）ことができる。よって、作業負担は大幅に軽減される。

【 0 0 9 3 】

【発明の効果】

本発明に係る請求項 1 に記載のプログラム開発手段及びプログラム実行手段を利用することにより、使用すべきデバイスドライバ（被制御対象オブジェクトソフトウェアモジュール）の内容を明確なものにすることができる。

【0 0 9 4】

本発明に係る請求項 2 に記載のプログラム開発手段及びプログラム実行手段を利用することにより、使用すべきデバイスドライバ（被制御対象オブジェクトソフトウェアモジュール）の内容を明確なものにすることができる。

【0 0 9 5】

本発明に係る請求項 3 に記載のプログラム開発手段及びプログラム実行手段を利用することにより、適切なデバイスドライバ（被制御対象オブジェクトソフトウェアモジュール）を正しく使用できる。また、プログラム開発手段及びプログラム実行手段において、デバイスドライバ（被制御対象オブジェクトソフトウェアモジュール）群を格納するメモリ領域を多く備える必要がない。

【0 0 9 6】

本発明に係る請求項 4 に記載のプログラム開発手段及びプログラム実行手段を利用することにより、適切なデバイスドライバ（被制御対象オブジェクトソフトウェアモジュール）を正しく使用できる。また、デバイスドライバ（被制御対象オブジェクトソフトウェアモジュール）を一元的に管理できる。さらに、プログラム開発手段及びプログラム実行手段において、デバイスドライバ（被制御対象オブジェクトソフトウェアモジュール）群を格納するメモリ領域を多く備える必要がない。

【0 0 9 7】

本発明に係る請求項 5 に記載のプログラム開発手段及びプログラム実行手段を利用することにより、開発作業において、プログラムの理解効率や作成効率が、向上する。

【0 0 9 8】

本発明に係る請求項 6 に記載のプログラム開発手段及びプログラム実行手段を利用することにより、デバイスを含む被制御対象オブジェクトの現況が的確に把

握できる。

【 0 0 9 9 】

本発明に係る請求項 7 に記載のプログラム開発手段及びプログラム実行手段を利用することにより、開発作業において、プログラムの理解効率や作成効率がより向上する。

【 0 1 0 0 】

本発明に係る請求項 8 に記載のプログラム開発手段及びプログラム実行手段を利用することにより、開発作業において、アイコンに対し様々な動作を設定できる。

【 0 1 0 1 】

本発明に係る請求項 9 に記載のプログラム開発手段及びプログラム実行手段を利用することにより、開発作業において、プログラムの理解効率や作成効率がより向上する。

【 0 1 0 2 】

本発明に係る請求項 1 0 に記載のプログラム開発手段及びプログラム実行手段を利用することにより、開発作業において、プログラムの理解効率や作成効率が更に向上する。

【 0 1 0 3 】

本発明に係る請求項 1 1 に記載のプログラム開発手段及びプログラム実行手段を利用することにより、プログラムのシミュレーションを、目視によりリアルタイムで確認できる。

【 0 1 0 4 】

本発明に係る請求項 1 2 に記載のプログラム開発手段及びプログラム実行手段を利用することにより、プログラムのモニタリングを、目視によりリアルタイムで確認できる。

【 0 1 0 5 】

本発明に係る請求項 1 3 に記載のプログラム開発手段及びプログラム実行手段を利用することにより、オブジェクト指向システム開発の考え方によってシステム開発を行うことができる。よって開発作業においてプログラムの理解効率や作

成効率を更に向上させることができる。

【 0 1 0 6 】

本発明に係る請求項 1 4 に記載のプログラム開発手段及びプログラム実行手段を利用することにより、プラグ・アンド・プレイ機能や活線抜粋の機能を充分に利用する自動機アプリケーションシステムの構築が可能になる。

【 0 1 0 7 】

本発明に係る請求項 1 5 に記載の記録媒体を利用することにより、自動機アプリケーションシステムの開発及び実行を適切に且つ効率よく行うことができる。

【図面の簡単な説明】

【図 1】 本発明の実施の形態 1 に従い自動機アプリケーションシステムを構築する手順の一例を示す図である。

【図 2】 プログラム作成手段、コントローラ、及び周辺デバイスの接続の例を示す、ブロック図である。

【図 3】 本発明の実施の形態 1 に係る、ソフトウェアアーキテクチャの例を示すブロック図である。

【図 4】 複数のデバイスが接続される、本発明の実施の形態 1 に係るソフトウェアアーキテクチャの例を示すブロック図である。

【図 5】 ソフトウェアモジュールデータベースの接続の形態を示す模式図である。

【図 6】 プログラム開発環境の模式図である。

【図 7】 プログラム開発例の図である。

【図 8】 本発明の実施の形態 2 に従い仮想的に自動機アプリケーションシステムを構築する手順の一例を示す図である。

【図 9】 仮想プログラム開発環境の模式図である。

【図 1 0】 プログラム開発例とシミュレーション例の図である。

【図 1 1】 バーチャルアシスタントに係るプログラム開発環境の模式図である。

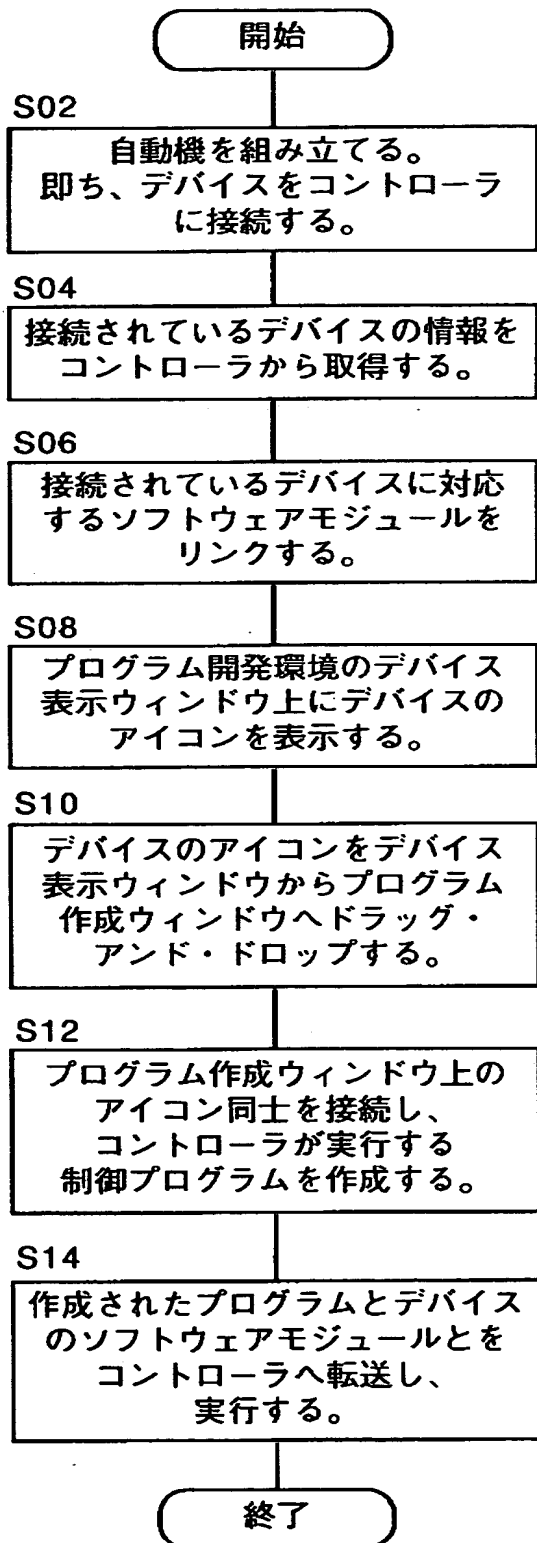
【図 1 2】 プログラム開発例とモニタリング例の図である。

【符号の説明】

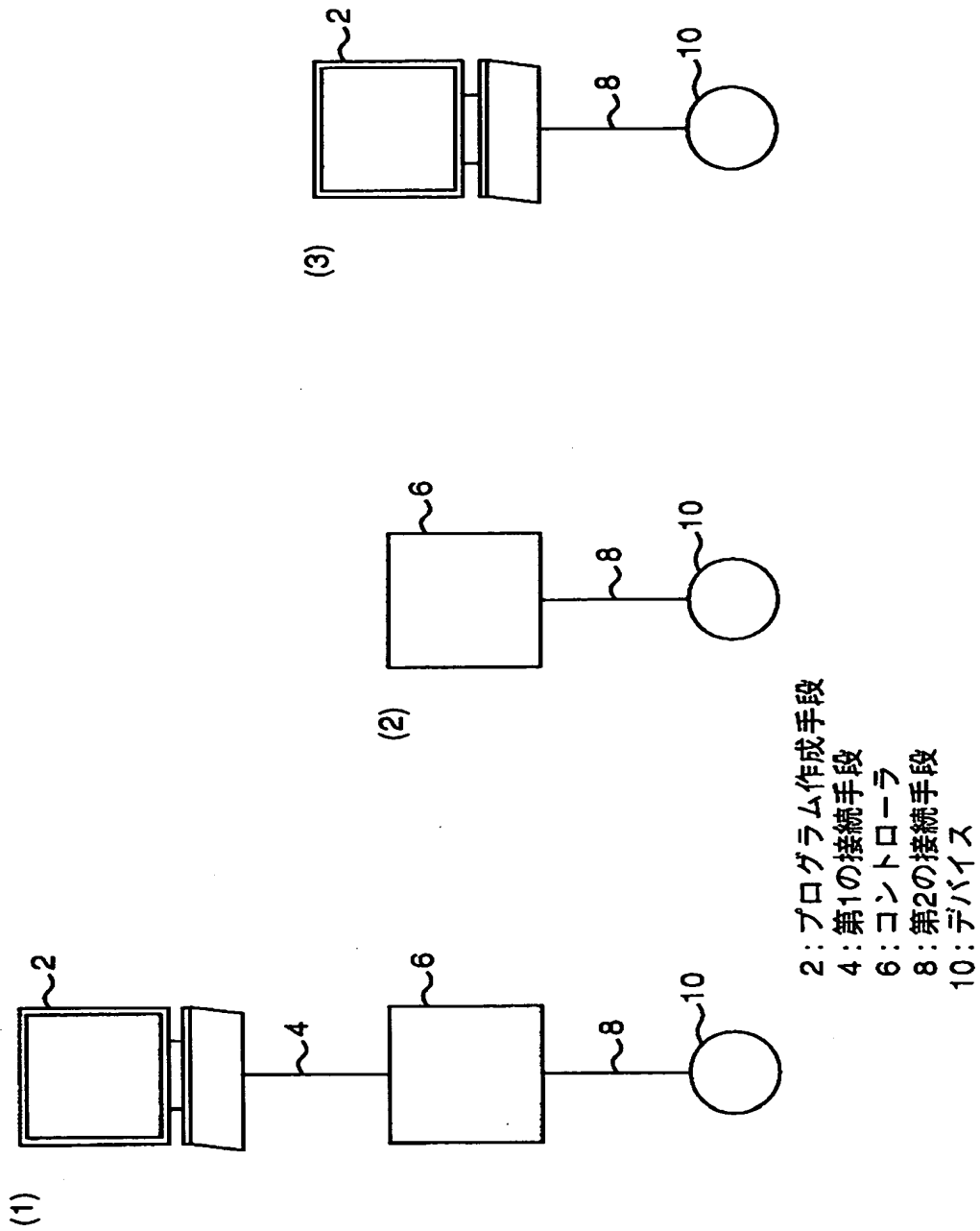
2 プログラム作成手段、4 第1の接続手段、6 コントローラ、8 第2の接続手段、10 デバイス、20 プログラム開発環境、22 プログラム実行環境、24 デバイスドライバ、26 通信手続き、40 ハードディスク、42 外部記憶媒体、44 サーバ・データベース記憶領域、46 伝送手段、60 プログラム開発環境画面、62 ツールバー、64 プログラム作成領域、66 デバイス表示領域、68、68'、68" デバイスアイコン、72 矢印、74 バーチャル管理人、76 条件分岐ブロック、78 ダイアログボックス（またはアプリケーションソフトウェア）、100 デバイスダイアログボックス、102 絵。

【書類名】 図面

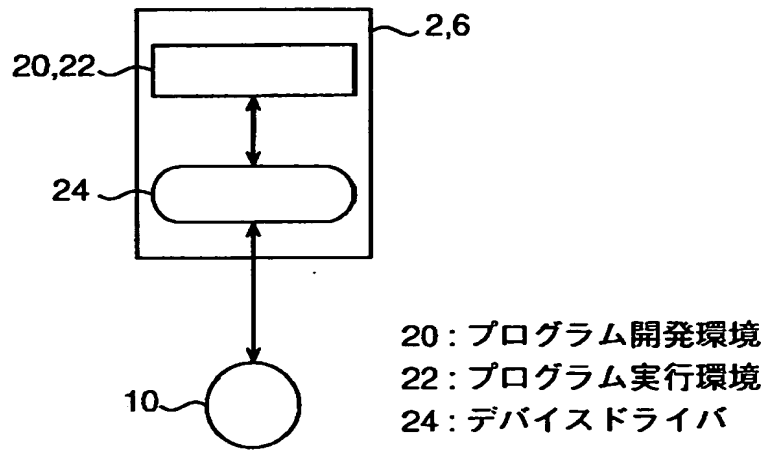
【図 1】



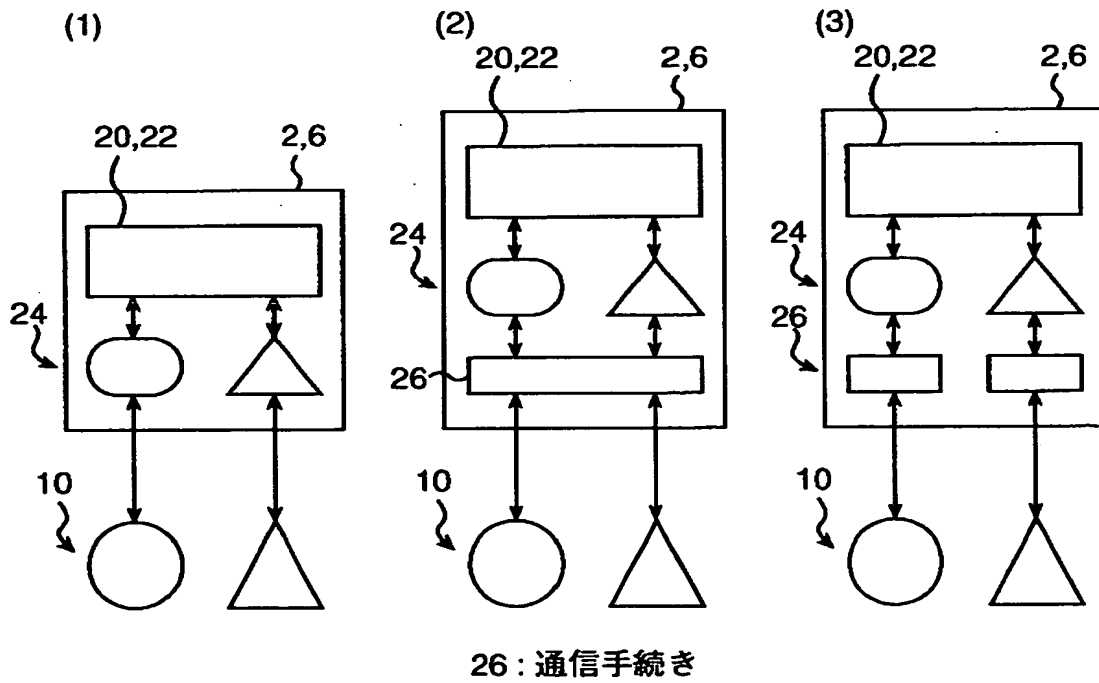
【図 2】



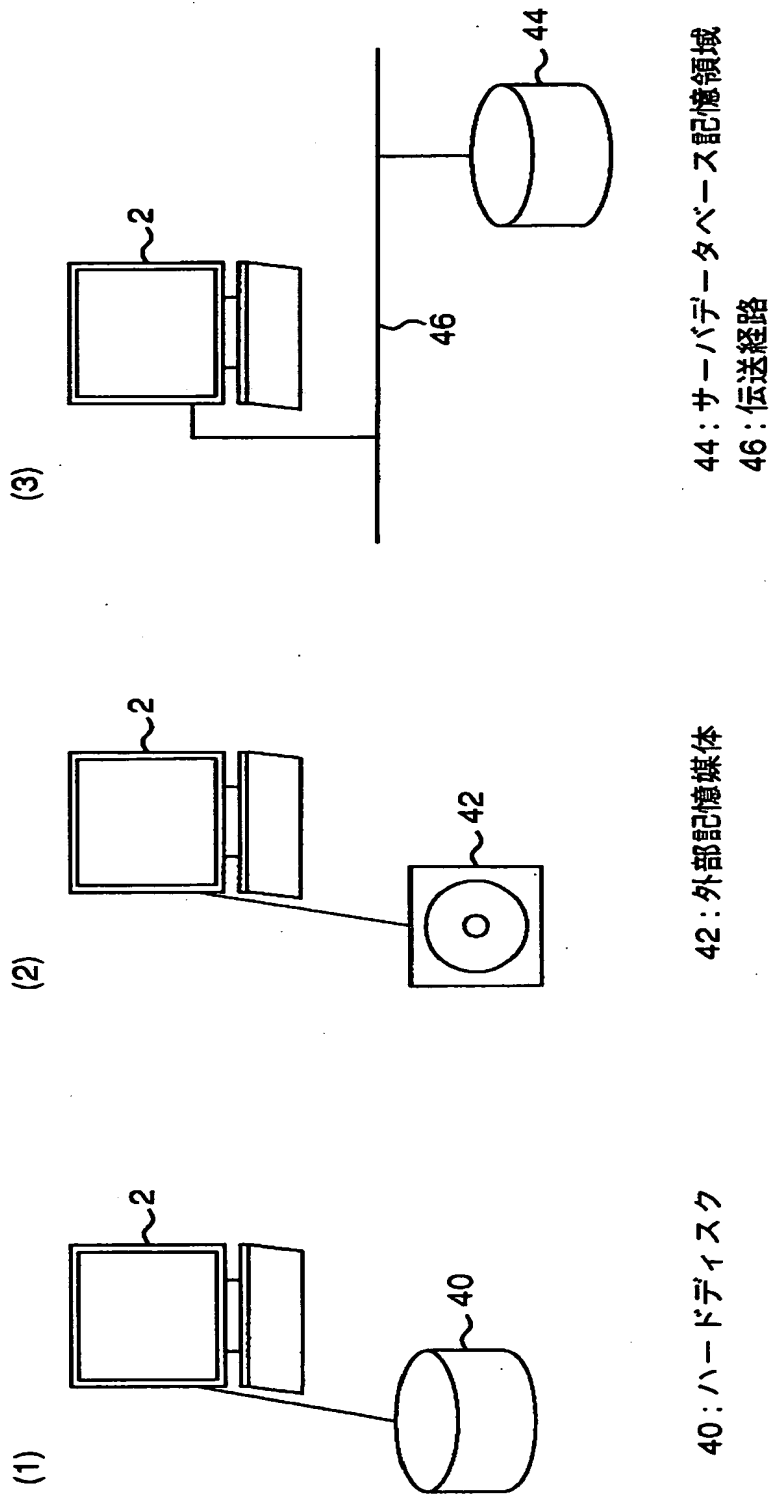
【図 3】



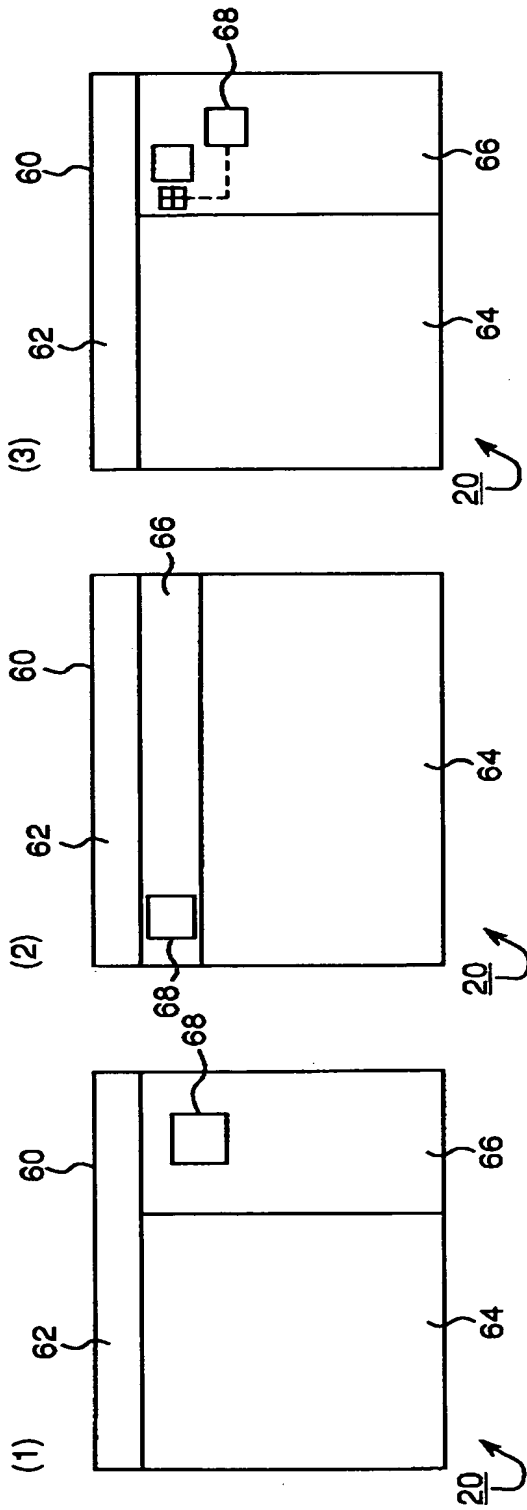
【図 4】



【図 5】



【図 6】



60: プログラム開発環境画面

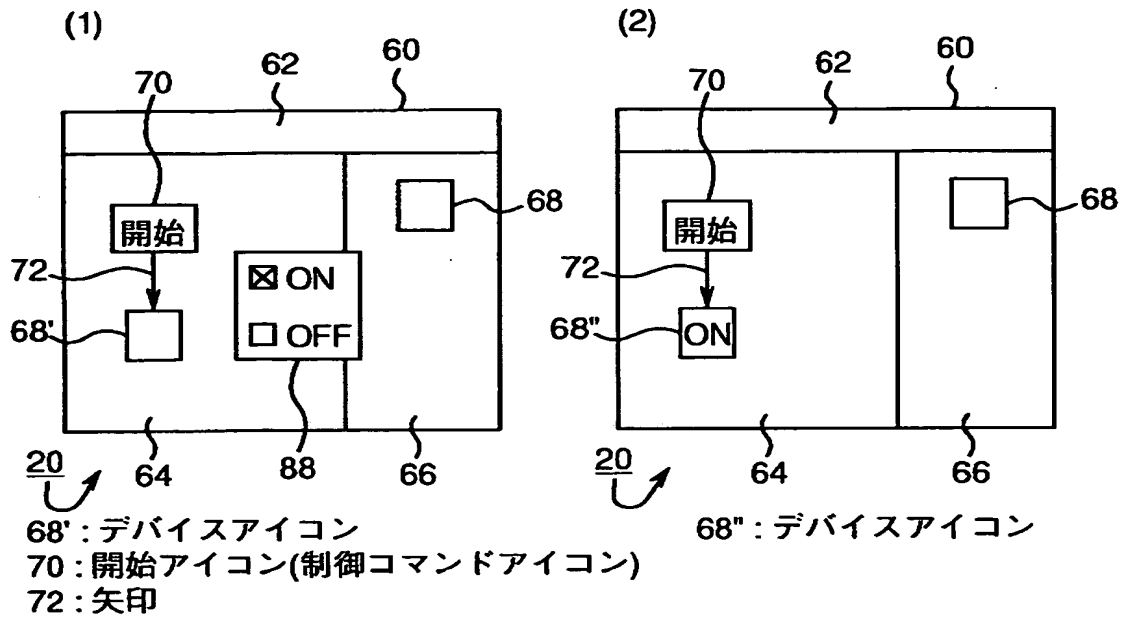
62: ツールバー

64: プログラム作成領域

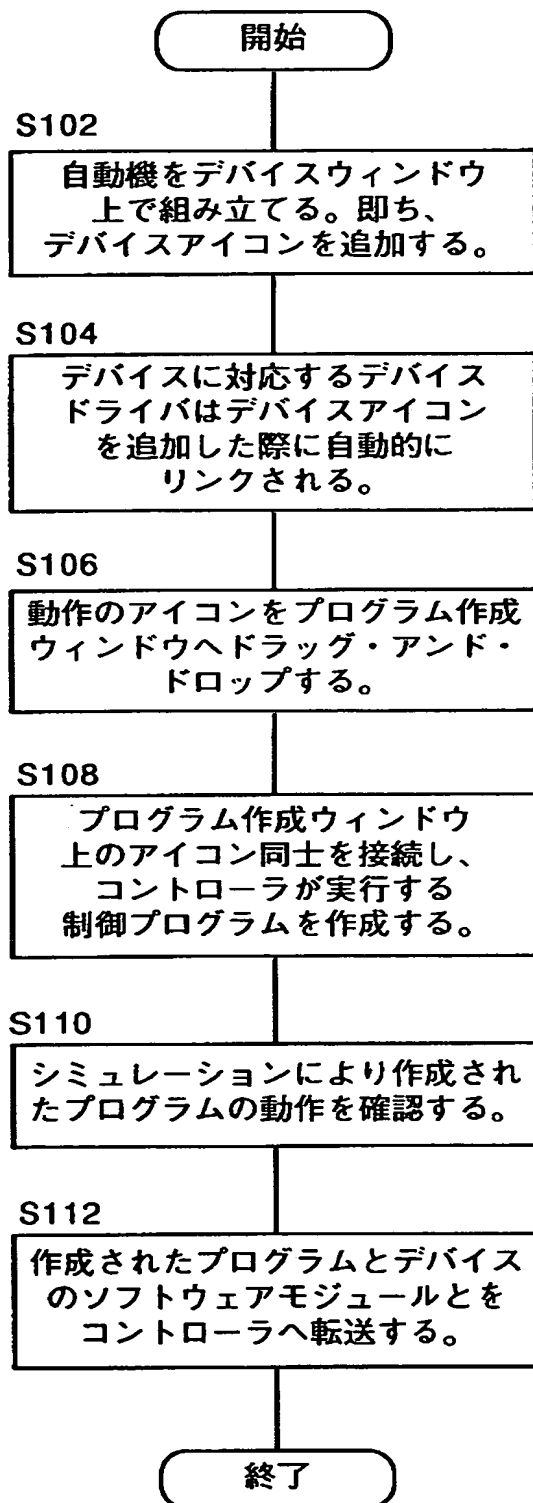
66: デバイス表示領域

68: デバイスアイコン

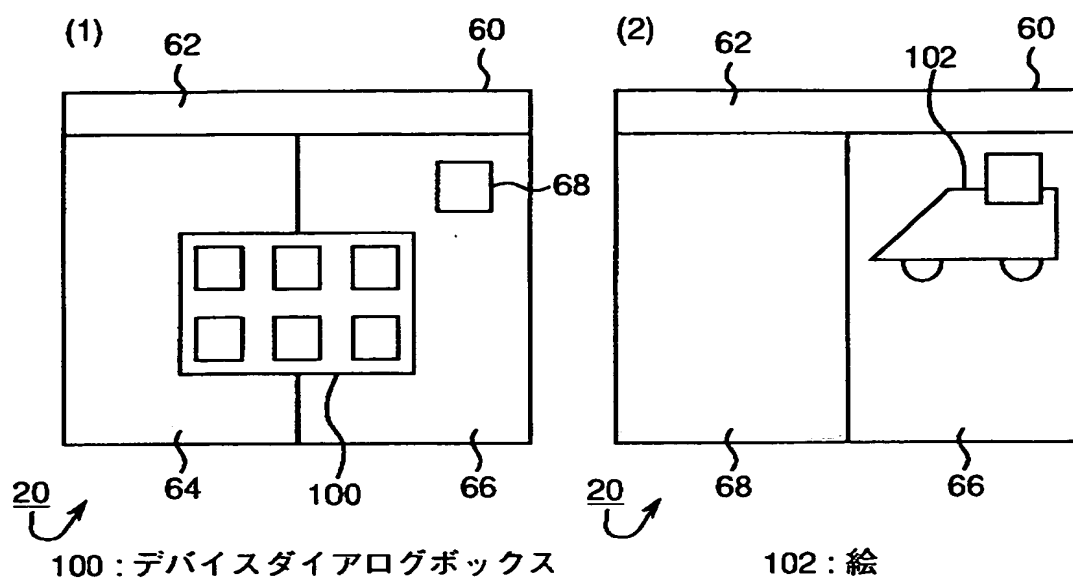
【図 7】



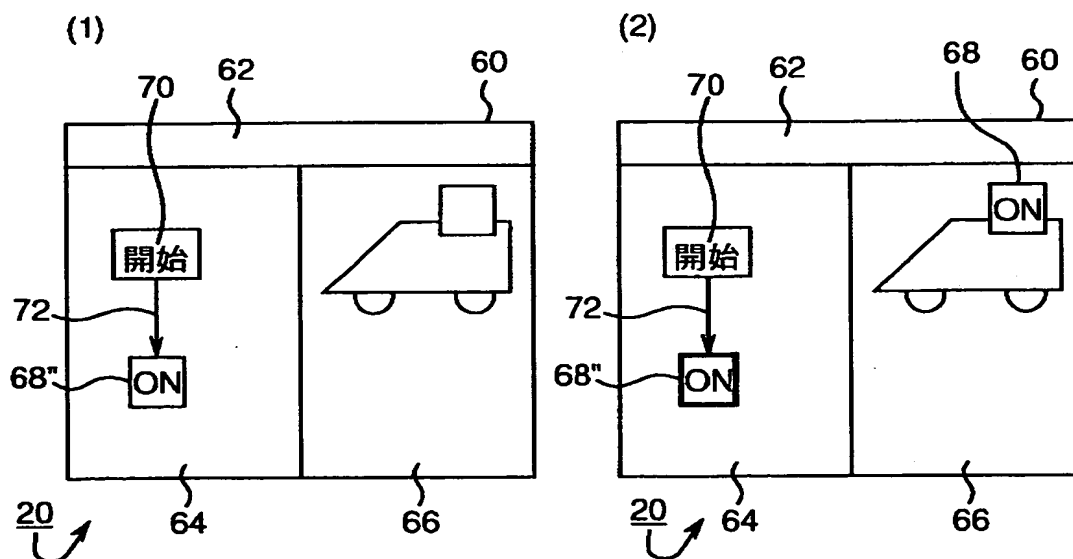
【図 8】



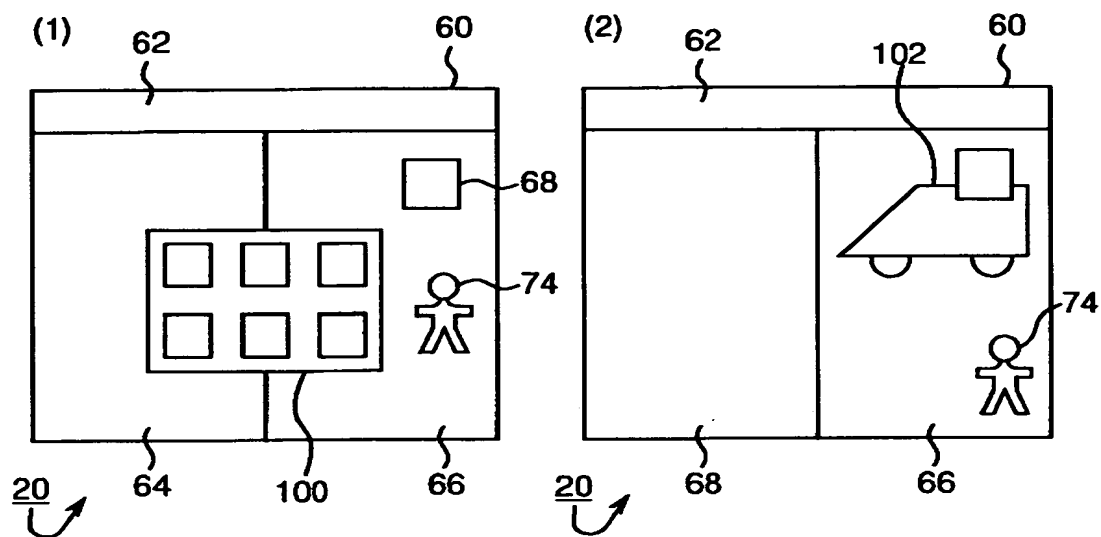
【図 9】



【図 1 0】

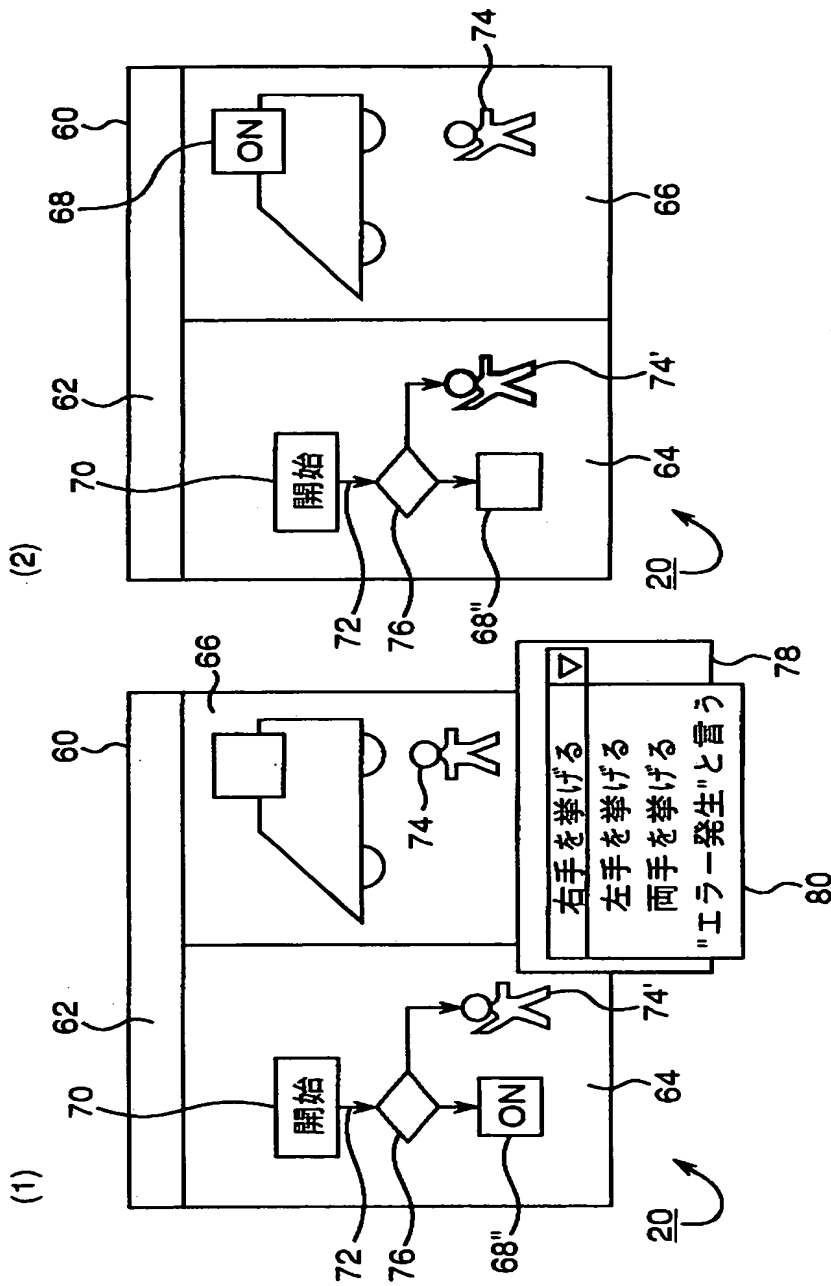


【図 1 1】



74 : バーチャル管理人

【図12】



76: 条件分岐
78: ダイアログボックス (またはアプリケーションソフトウェア)

【書類名】 要約書

【要約】

【課題】 自動機を構築する際に、デバイスやコントローラに係るセットアップの手間を大幅に削減するとともに、各種制御プログラムの作成を簡素化する

【解決手段】 デバイスを接続すれば必要ソフトウェアモジュールを自動的にリンクする機能を備える。デバイスアイコンや制御コマンドアイコンのダイアログボックス等、プログラミングのための各種要素を、デバイスに係る必要ソフトウェアモジュールに含める。各種制御プログラムの作成を、目的のデバイスアイコンを中心とする「デバイス指向プログラミング」により行ない、作成作業の簡素化を目指す。

【選択図】 図 1

出 願 人 履 歴 情 報

識別番号 [000006013]

| | |
|----------|-------------------|
| 1. 変更年月日 | 1990年 8月24日 |
| [変更理由] | 新規登録 |
| 住 所 | 東京都千代田区丸の内2丁目2番3号 |
| 氏 名 | 三菱電機株式会社 |